UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/729,666	12/05/2003	Jason D. Sollom	RA 5512(1056-004US01)	3160

7590 05/03/2006

Charles A. Johnson
Unisys Corporation
P O Box 64942 MS 4773
St. Paul, MN 55164

EXAMINER	
GEBRESILASSIE, KIBROM K	

ART UNIT	PAPER NUMBER
2128	

DATE MAILED: 05/03/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/729,666	SOLLOM ET AL.	
	Examiner	Art Unit	
	Kibrom K. Gebresilassie	2128	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *05 December 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *05 December 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the amended application filed on December 5, 2003.

2. Claims 1-30 have been examined and rejected.

Oath/Declaration

3. The Office acknowledges receipt of properly signed oath/declaration filed December 5, 2003.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

The factual inquiries set forth in *Graham* **v.** *John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

5. Claims 1-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Stuart Oberman et al. US 2001/0054140 A1, herein referred as **Oberman**, in view of US 2004/0230949 A1 issued to Talwar et al, herein referred as **Talwar.**

As per Claim 1:

Oberman discloses a processor-based method performed by software emulating an instruction processor, the method comprising: processing read instructions with an emulated processor to output independent read requests via an operand interface and an op-code interface of the emulated processor (Abstract lines 1-3); independently comparing op-code reference data and operand reference data to operands and op-codes received in response to the read requests ([0079]); and recording results of the independent comparisons ([0080] and table 1).

Oberman fails to disclose an emulation environment.

Talwar discloses an emulation environment ([0025], Fig. 1A).

It would have been obvious to one of ordinary skill in the art at the time of the invention was made to combine the teachings of Oberman related to a multimedia execution unit incorporated within a microprocessor for accommodating high-speed multimedia applications with the teachings of Talwar related to creating emulation language code and native language code associated with the emulation language code. The motivation for doing so would have been more convenient to verify and load native language information using components of emulation verification systems [0027]. Hence a skilled artisan having access to the teaching of Oberman and Talwar would have knowingly modified the teaching of Oberman with Talwar.

As per Claim 2:

Oberman discloses storing the op-code reference data and the operand reference data within a set of data memories of the emulated instruction processor

([0010] lines 5-14); maintaining within the emulated instruction processor an operand data pointer to address the operand reference data and an op-code pointer to address the op-code reference data; and independently accessing the operand reference data with the operand data pointer and the op-code reference data with the op-code data pointer during processing of the read instructions to verify the received op-codes and the received operands (Abstract lines 1-3).

As per Claim 3:

Oberman discloses when processing the read instructions, determining whether each of the read instructions required an operand read request or an op-code read request; and independently updating the op-code reference pointer or the operand reference pointer based on the determination ([0076]).

As per Claim 4:

Oberman discloses the read instructions form part of an instruction stream executed by the emulated instruction processor, the method further comprising: processing a flow control instruction of the instruction stream with the emulated instruction processor; and upon processing the flow control instruction, synchronizing the op-code reference pointer and the operand reference pointer to respectively address a portion of the op-code reference data and a portion of the operand reference data associated with a target address of the flow control instruction ([0010] lines 21-31).

As per Claim 5:

Oberman discloses, compiling test software to output the operand reference data, the op-code reference data, and the instruction stream ([0009]).

As per Claim 6:

Oberman discloses independently comparing further comprises: latching the op-code reference data within a first latch within the emulated instruction processor (Fig. 8 element 720); and comparing the latched op-code referenced data and the received op-codes with a comparator to produce the results ([0079]).

As per Claim 7:

Oberman discloses latching the operand reference data within a first latch within the emulated instruction processor (Fig. 8 elements 710A or 710B); and comparing the latched operand referenced data and the received operand with a comparator to produce the results ([0079]).

As per Claim 8:

Oberman discloses storing write data within a data memory of the emulated instruction processor; maintaining within the emulated instruction processor a write data pointer to address the write data; and processing a write instruction with the emulated processor to output a mite request via a data interface of the emulated processor, wherein the write request comprises a portion of the write data referenced by the write pointer ([0080], [0081]).

As per Claim 9:

Oberman discloses generating a report based on the results, wherein the report identifies any of the received op-codes that do not match the op-code reference data and any of the received operands that do not match the operand reference data ([0080], and Table 1).

As per Claim 10:

Oberman discloses storing addresses associated with received operands and the op-codes, copies of the received operands or op-codes, copies of the reference operands and the reference op-codes, or copies of the instruction ([0013] lines 3-6).

As per Claim 11:

Oberman discloses recording the results comprises recording the result within a register within the emulated instruction processor ([0080] and table 1).

As per Claim 12:

Oberman discloses applying bit masks to at least a portion of a result of the independent comparisons of the op-code reference data and the operand reference data to the op-code and the operand received in response to the read requests ([0079] and [0007] lines 30-33).

As per Claim 13:

Oberman discloses storing the addresses of a small number of received operands and received op-codes within a within the emulated instruction processor; and selectively enabling and disabling access to the cache when executing subsequent read instructions and waiting to output read requests via the operand interface and the op-code interface when the read instructions request operands and op-codes are invalidated within the cache based on a configurable option ([0013]).

As per Claim 14:

Oberman discloses a processor-based system for emulating an instruction processor comprising: a computing system to provide an emulation environment

([0025], [0026], Fig. 1); and an instruction processor having an operand interface and an op-code interface (Abstract lines 1-7), wherein the software emulates the instruction processor by processing read instructions and outputting corresponding read requests on the operand interface or the op-code interface ([0009]), and independently comparing op-code reference data and operand reference data to operands and op-codes received from the operand interface and op-code interface in response to the read requests ([0079]);.

Oberman fails to disclose an emulation environment.

Talwar discloses an emulation environment ([0025], Fig. 1A).

It would have been obvious to one of ordinary skill in the art at the time of the invention was made to combine the teachings of Oberman related to a multimedia execution unit incorporated within a microprocessor for accommodating high-speed multimedia applications with the teachings of Talwar related to creating emulation language code and native language code associated with the emulation language code. The motivation for doing so would have been more convenient to verify and load native language information using components of emulation verification systems [0027]. Hence a skilled artisan having access to the teaching of Oberman and Talwar would have knowingly modified the teaching of Oberman with Talwar.

As per Claim 15:

Oberman discloses software emulates the instruction processor by recording results of the independent comparisons within a register of the emulated instruction processor ([0080] and table 1).

As per Claim 16:

Oberman discloses the emulated instruction processor comprises: a first data memory to store the op-code reference data; and a second data memory to store the operand reference data ([0013] lines 3-6).

As per Claim 17:

Oberman discloses the emulated instruction processor comprises: a control unit that maintains an operand data pointer to address the operand reference data within the first data memory and an op-code pointer to address the op-code reference data within the second data memory, wherein the control unit independently accesses the operand reference data with the operand data pointer and the op-code reference data with the op-code data pointer during processing of the read instructions to verify the received op-codes and the received operands ([0080], [0081]).

As per Claim 18:

Oberman discloses upon processing the read instructions, the control unit determines whether each of the read instructions required an operand read request or an op-code read request, and independently updates the op-code reference pointer or the operand reference pointer based on the determination ([0076]).

As per Claim 19:

Oberman discloses the read instructions form part of an instruction stream executed by the emulated instruction processor, and upon processing a flow control instruction of the instruction stream, the control unit synchronizes the op-code reference

pointer and the operand reference pointer to respectively address a portion of the op-code reference data and a portion of the operand reference data associated with a target address of the flow control instruction ([0010] lines 21-31).

As per Claim 20:

Oberman discloses a compiler executing on the computing system to compile test software to output the operand reference data, the op-code reference data, and the instruction stream for execution by the emulated instruction processor ([0009]).

As per Claim 21:

Oberman discloses the emulated instruction processor further comprises: a latch to latch the op-code reference data from the first data memory (Fig. 8 Unit 20); and a comparator to compare the latched op-code referenced data and the received op-codes ([0079]).

As per Claim 22:

Oberman discloses the emulated instruction processor further comprises: a latch to latch the operand reference data from the second data memory (Fig. 8 Unit 36F); and a comparator to compare the latched operand referenced data and the received operands ([0079]).

As per Claim 23:

Oberman discloses the emulated instruction processor further comprises: a data memory to store write data; and a control unit to maintain a write data pointer to address the write data wherein the control unit processes a write instruction to output a write request via a data interface of the emulated processor, and further wherein the control

unit generates the write request to comprises a portion of the write data referenced by the write pointer ([0080], [0081])..

As per Claim 24:

Oberman discloses emulation control software executing on the computing system to generate a report that presents/identifies any of the received op-code that do not match the op-code reference data and any of the received operands that do not match the operand reference data ([0080], and Table 1).

As per Claim 25:

Oberman discloses a set of memories to store bit masks ([0013] lines 3-6); and bit masks to compare results of the comparison of the received operands and the received op-codes to the reference operands and the reference op-codes to mask portions of these comparisons ([0007] lines 30-33 and [0079]).

As per Claim 26:

Oberman discloses a cache to store the addresses of the received operands and received op-codes, wherein the emulated instruction processor selectively waits to issue read requests for subsequent read instructions via the operand interface and the op-code interface until the read instructions request operands and op-codes become invalidated within this cache based on a configurable option ([0013]).

As per Claim 27:

Oberman discloses a processor-based system for emulating an instruction processor comprising: compiling means for compiling test software to produce operand reference data, op-code reference data, and an instruction stream having read

instructions ([0010] lines 1-9); and having an operand interface and an op-code interface ([0010] lines 1-5), wherein the emulating means comprises: controlling means for controlling the emulated instruction processor to process the read instructions and output corresponding read requests on the operand interface or the op-code interface ([0014] lines 17-27), receiving means for receiving operands and op-codes from the operand interface and op-code interface in response to the read requests (Fig. 9 element 710A and 720), and comparing means for independently comparing the op-code reference data and the operand reference data to the received ([0079]).

Oberman fails to disclose emulating means for emulating an instruction processor.

Talwar discloses emulating means for emulating an instruction processor ([0025], Fig. 1A).

It would have been obvious to one of ordinary skill in the art at the time of the invention was made to combine the teachings of Oberman related to a multimedia execution unit incorporated within a microprocessor for accommodating high-speed multimedia applications with the teachings of Talwar related to creating emulation language code and native language code associated with the emulation language code. The motivation for doing so would have been more convenient to verify and load native language information using components of emulation verification systems [0027]. Hence a skilled artisan having access to the teaching of Oberman and Talwar would have knowingly modified the teaching of Oberman with Talwar.

As per Claim 28:

The limitation of claim 28 has already been discussed in the rejection of Claim 16. It is therefore rejected under the same rationale.

As per Claim 29:

Oberman discloses first referencing means for addressing the operand reference data within the first storing means; and second referencing means for addressing the op-code reference data within the second storing means ([0013] lines 3-6).

As per Claim 30:

The limitation of claim 30 has already been discussed in the rejection of Claim 9. It is therefore rejected under the same rationale.

Conclusion

6. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

US Patent No. 6,401,155 issued to Saville et al.

US Patent No. 6,647,489 issued to Col et al.

Any inquiring concerning this communication or earlier communication from the examiner should be directed to Kibrom K. Gebresilassie whose telephone number is (571) 272-8571. The examiner can normally be reached on Monday-Friday, 8:30 am to 4:30 pm. If attempts to reach the examiner by telephone are unsuccessful, the examiner supervisor, Kamini shah can be reached at (571) 272-2279. The official fax number is (571) 273-8300. Any inquiring of a general nature relating to the status of this application should be directed to the group receptionist whose telephone number is (571) 272-3700.

Kibrom K. Gebresilassie
Patent Examiner
U.S. Patent and Trademark Office
Simulation and Emulation, Art Unit 2128
401 Dulany St., Room 5C25 (Randolph)
Alexandria, VA 22314-5774
Tel: 571-272-8571
Kibrom.gebresilassie@uspto.gov